Volume

4

FLEXCEL STUDIO FOR VCL AND FIREMONKEY

TMS Software



Exporting Excel files to PDF

Table of Contents

TABLE OF CONTENTS ..... **I**
INTRODUCTION **1**
CREATING PDF FILES **1**
INTRODUCTION: 1
USING TPDFWRITER 1
USING FLEXCELPDFEXPORT 2
PREPARING A FILE FOR PRINTING **3**
FONT MANAGEMENT **4**
SELECTING WHICH FONTS TO USE 4
ACCESSING TRUE TYPE DATA. 5
FONTS AND EXCEL 2007 ..... **6**
SETTING THE "NORMAL" FONT 7
DEALING WITH MISSING FONTS AND GLYPHS 8
Problem 1: Missing fonts *9*
Problem 2: Missing Glyphs *10*
Problem 3: Faux Italics and Bolds *10*
SIGNING PDF FILES **12**
EXPORT TO PDF AND FLEXCEL RECALCULATION **13**
USING FLEXCELPDFEXPORT IN OSX AND IOS **13**

Introduction

FlexCel comes with a full PDF writer that allows you to natively export Excel files to PDF, without needing to have Acrobat or Excel installed. While the output will not be exactly the same as Excel, a lot of effort has been done to make it as similar as possible.

Creating PDF files

Introduction:

FlexCel provides two ways to create a PDF file. At higher level, you can use TFlexCelPdfExport component to natively convert an Excel file into pdf. At the lower level, you have the TPdfWriter class, that provides a primitive API to create pdf files.

Using TPdfWriter

TPdfWriter is the low level option, and it was really not designed to be used directly, but to provide the methods TFlexCelPdfExport needs to do its job.

But, it can be used standalone to create a PDF file for scratch, or most likely to modify the output from TFlexCelPdfExport using one of the Before/After CreatePage events.

We will not cover it in detail here since methods are documented in FlexCel.chm, but we will mention that you can find an example in how to use TPdfWriter in the API Demos.

Using FlexCelPdfExport

This is the high level option, and the one you would normally use. To export an xls file to pdf, you need to:

1. Create a TFlexCelPdfExport in your code.
2. Set the FlexCelPdfExport properties. You will find there are not a lot of properties (things like margins, printing gridlines or not, etc) and this is because all this information is read from the Excel file. If you need to change them, change the associated properties on the attached TXlsFile.
3. Attach an TXlsFile to the TFlexCelPdfExport component, by assigning :

```
flexCelPdfExport1.Workbook = someXlsFile.
```

4. Export the file. You can use .Export to export only the active sheet, or BeginExport/ExportSheet/EndExport to export more than one.

Preparing a file for printing

Excel printing is "printer dependent", that is, it can print different things to different printers.

A rectangle printed to a standard laser printer can be 11 x 11 cm. and the same rectangle printed to a dot matrix printer might be 11.5 x 11.5 cm. As you can imagine, this might prove problematic when your file is designed to be printed in one page at 100% zoom. In some printers, it might actually print in one page, but in others it might print in two, or even in 4.

In the example above, if your file was 11x11 and you were printing in an 11x11 page, it will fit in one page. But if the printer prints a little bigger, 11.5, it will actually need 4 pages to fit.

Different from Excel, FlexCel printing is normally resolution independent (using GDI+ or FireMonkey) and it will print the same size everywhere. So when we calibrated the printing, we had to choose a mix of all possibilities. If Excel prints a rectangle 10cm. wide in one printer, 11 in other and 12 in other, FlexCel will print the same in all of them. We settled by emulating printing in a 600 dpi laser printer, but even this is not 100% exact. Laser printers don't print exactly the same either in Excel. In short, FlexCel results might not be the same size as Excel results for your particular printer.

So whenever you need to "fit" into a page, it is recommended that you use the "Print to Fit" setting in Excel. Look at the "Preparing for printing" section in the API guide for details on this. But if you don't want to use print to fit, make sure you leave a large enough margin at the right and at the bottom, so the page prints fine everywhere, including FlexCel.

Font Management

Due to some framework limitations the font handling can be problematic, so here we discuss some concepts that might help with the issue.

Selecting which fonts to use

First of all, you need to know that there are two different kinds of fonts supported by FlexCel's TPdfWriter.

1. **PDF internal fonts**. PDF defines 14 standard fonts that must be available to any PDF viewer, so you don't need to embed the fonts on the PDF document. They will always be supported by the PDF viewer.

   Those fonts include a Serif (Times New Roman-like), a Monospace (Courier-like) and a Sans Serif (Arial-like) alternatives, on four variants each (regular, bold, italic and bold italic) and two Symbol fonts.

2. **True Type fonts**. Those are standard Windows fonts.

   When exporting to PDF, you can chose between three different ways to handle fonts, depending on the value you set on the FontMapping property:

   a. **ReplaceAllFonts**. This will replace all fonts on the xls file to the most similar ones on the 14 standard fonts. This way you get the minimum file size and the maximum portability, but the exported PDF file might not look exactly the same, as you will lose all fancy fonts.

   b. **ReplaceStandardFonts**. This is a compromise solution. It will only replace Arial, Times New Roman and Courier for the standard fonts, and use True type for all the others. You will get a bigger PDF file (if you embed the true type fonts), but it will look as the xls file.

   c. **DoNotReplaceFonts**. This will only use true type fonts. It will be the one that better matches the xls file, but it will be a lot larger (if you embed the true type fonts) or might not look good when the user does not have the fonts you used installed (if you don't embed them)

Also, you can choose whether to embed the True Type fonts or not. If you embed them, the file will be bigger, but also will render well when the user does not have the fonts on his machine. To avoid issues, it is normally recommended that you embed all fonts.

Note that if you use Unicode characters, the fonts will always be embedded no matter which embed option you choose, because this is needed to ensure the Unicode mapping will remain correct.

Accessing True Type data.

In order to embed fonts, and get many font metrics that we need to create the pdf file, FlexCel needs to access the raw *.ttf files directly. And to find those files, you need to tell FlexCel where the font folder is.

If you do nothing, FlexCel will look at the default font folder in either Windows or OSX.

If this folder does not exist, FlexCel will search on "**<folder where FlexCel.dll is>/Fonts**". If you are running OSX, you might create a ***symbolic link*** to the place where the fonts are installed.

As a last resort, FlexCel provides a **OnGetFontFolder** event that allows you to specify where fonts are stored on your system. Here you can tell FlexCel where the fonts are.

Avoid using this event if you can, since when you use it your code will not transparently run on OSX and Windows, you will have to have two different binaries. A Symbolic link from the FlexCel installation folder to the fonts folder should be a more elegant solution.

Of course, you will not face any of these problems when using only standard PDF fonts (and no Unicode)

Fonts and Excel 2007

Excel 2007 changed the default font in a new file to be "Calibri" instead of "Arial". This might bring you problems if you develop in a machine that has Excel 2007 installed, but you deploy in a server that doesn't. There are two solutions for this:

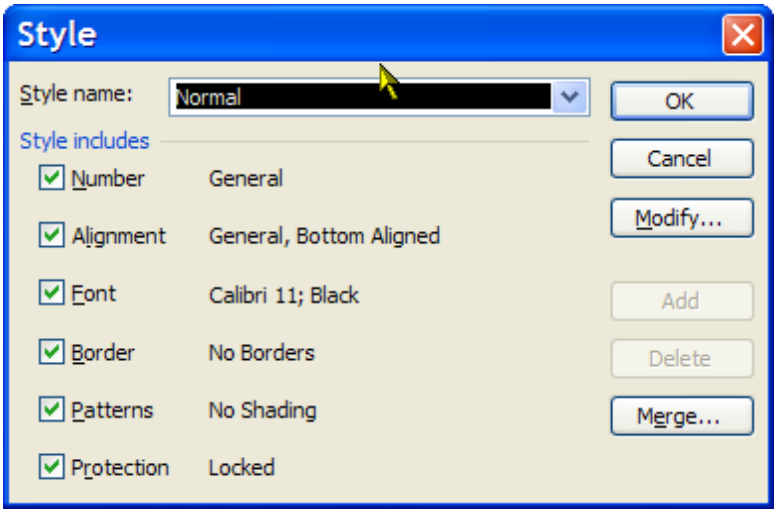
1. You can **copy the Excel 2007 fonts to the server**, and make sure you **embed the fonts in the PDF file**. Note that if you do not embed the fonts, any user who does not have Excel 2007 will not be able to see your PDF file correctly.

2. If you want maximum portability, make sure you change all fonts to Arial or Times new Roman in your template before exporting.

Setting the "Normal" font

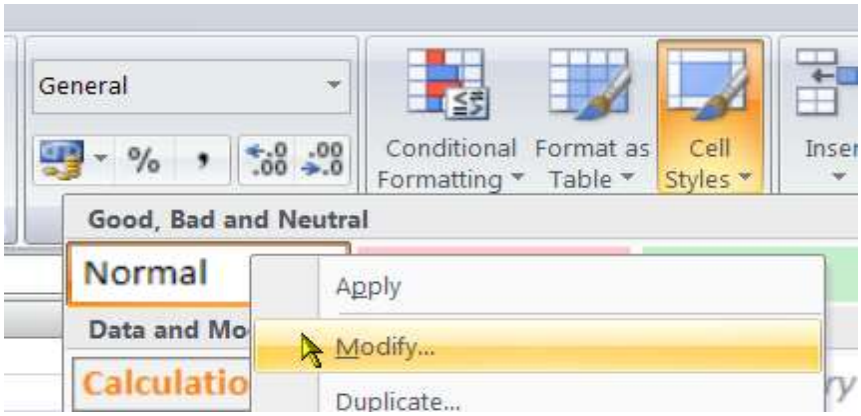
Especially important when changing the fonts is to make sure the "Normal" format uses a known font. You can see/change the normal font this way:

1. In **Excel 97-2003**: Go to "Menu->Format->Style..." You should see the following dialog:



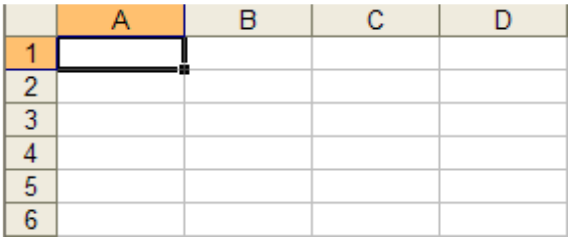
Verify that the "Normal" style has a known font like "Arial"

2. In **Excel 2007 or newer**: In the home tab in the Ribbon, select Cell Styles, right click in "Normal" and choose modify. Note that depending on your screen resolution, "Cell Styles" might show as a button or just display the "Normal" box directly in the ribbon.

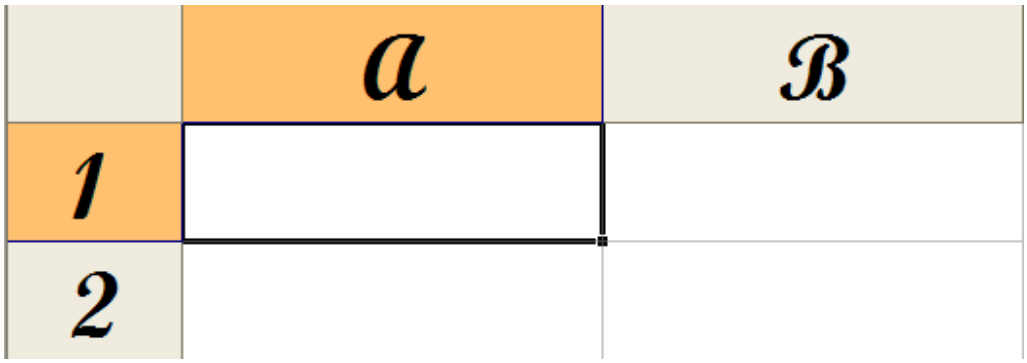


You will get a similar dialog to the one you got in Excel 2003. Again, make sure the normal style uses a font you have in your server.

The "Normal" style is used not only for empty cells, but to set the column widths. For example, this is how an empty sheet looks with "Normal" style using Arial 10:



And this is how it looks using Script MT 36:



As you can see, the font used in the "Normal" style is used to draw the headings "A", "B", "1", etc., and even more important, it is used to calculate the column width. Column with is measured as a percentage of the "0" character width in the normal font. If you change the normal font, column widths will change.

If you do not have the "Normal" font installed in your server, Windows will replace it with a substitute, and it will probably have a different width for the "0", leading to a wrong column width. So it is important that you have that font installed in your server.

Dealing with missing fonts and glyphs

There are three main font-related problems you might find when converting an xls file to PDF, and we are going to cover them in this section. The errors are non fatal, and that means that the file will be generated anyway, but it will not look as good as it could.

You can control what to do when any of these errors happen by hooking an event to the T**FlexCelTrace** static class. From this event, you could write a log file when any of these errors happen, warn the user, or just raise an exception if you want to abort file generation.

Problem 1: Missing fonts

This is normally the easiest one to solve, and normally happens when deploying an application to a server. As explained in the section above, this often happens with "Calibri" font that gets installed by Excel 2007, and probably will not be installed in the server. As FlexCel needs the font to be present in order to create the pdf file, it will substitute it with a "similar" font, normally Arial or Microsoft sans serif.

This might not be an issue if there are any fonts in the system that are similar to the one being replaced, but it can be a big issue with Calibri, since that font has very different metrics from the font it gets replaced (Arial). As an example, here you can see an Excel 2007 exported to PDF in a machine that has Calibri installed an in other that doesn't:

With Calibri installed in the fonts folder:

	A	B	C	D
1				
2		This text is in Calibri	Balance of Year	Balance Sheet
3		**This text is in Arial**	**Balance**	**Balance Sheet**
4		Assets		
5		Current assets:	2007	2008
6		Cash	-	-
7		Investments	-	-

Without Calibri installed (Replaced by Arial):

	A	B	C	D
1				
2		**Calibri**	**Year**	**Balance Sheet**
3		**This text is in Arial**	**Balance**	**Balance Sheet**
4		**Assets**		
5		**Current assets:**	**2007**	**2008**
6		Cash	-	-
7		Investments	-	-
8		Inventories	-	-

As you can see in the images, Calibri is much narrower than Arial, so the text in cell B2 "This Text is in Calibri" is cut and only "Calibri" shows in the second screenshot. If you are seeing

lots of cut text in the server while the files are exported fine in your development machines, this is probably the cause.

The solution for this problem is easy; make sure you have all the fonts you use installed in your system. If you want to get notified whenever this automatic font replace happens, you can catch the "FlexCelError.PdfFontNotFound" errors in TFlexCelTrace, and use it to notify the user he should install the missing fonts.

Problem 2: Missing Glyphs

This problem happens when you are using a font that doesn't contain the character you want to display. If you for example write

```
"日本 に行きたい。"
```

inside a cell and keep the font "Arial", you will see the correct characters in Excel, but when exporting you might see blank squares like this:

```
□□ □□□□□□
```

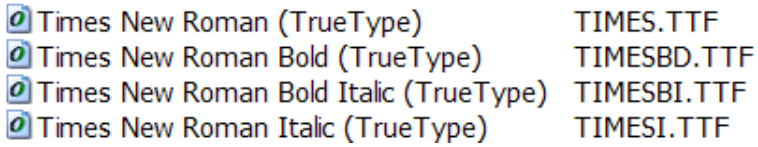
The reason for this is that "Arial" doesn't actually contain Japanese characters, and Excel is "under the hood" using other font (normally MS Mincho) to display the characters. To emulate this behavior, FlexCel provides a "FallbackFonts" property, where you can enter a list of fonts to try if the font that was supposed to be used doesn't have the character. If no font in the FallbackFont chain contains the glyph, you will see a blank square.

The solution in this case is to use fonts that actually have the characters you want to display, or ensure that some fonts in the FallbackFonts properties have them. By default FlexCel uses "Arial Unicode MT" as a fallback font, but you can add as many others as you need.

If you want to get notified when this happens so you can warn the user to change the fonts, you can catch the "FlexCelError.PdfGlyphNotInFont" and "FlexCelError.PdfUsedFallbackFont" errors in TFlexCelTrace.

Problem 3: Faux Italics and Bolds

The last problem is related to fonts that don't have a specific "Italic" or "Bold" variant. Normally, a quality font comes with four files including four variants of the font: Bold, Italic and BoldItalic. If you look in your font folder, you will see things like this:



That is, a different file for each font variant. If the font comes with only one "Normal" file, the italics can be faked by the system by slanting the characters, and bold can be faked by adding some weight to the characters. But of course this leads to low quality results, and should be avoided whenever possible.

There is other problem with "fake" Italics and Bold, and that is that Acrobat will not show them when you embed the fonts.

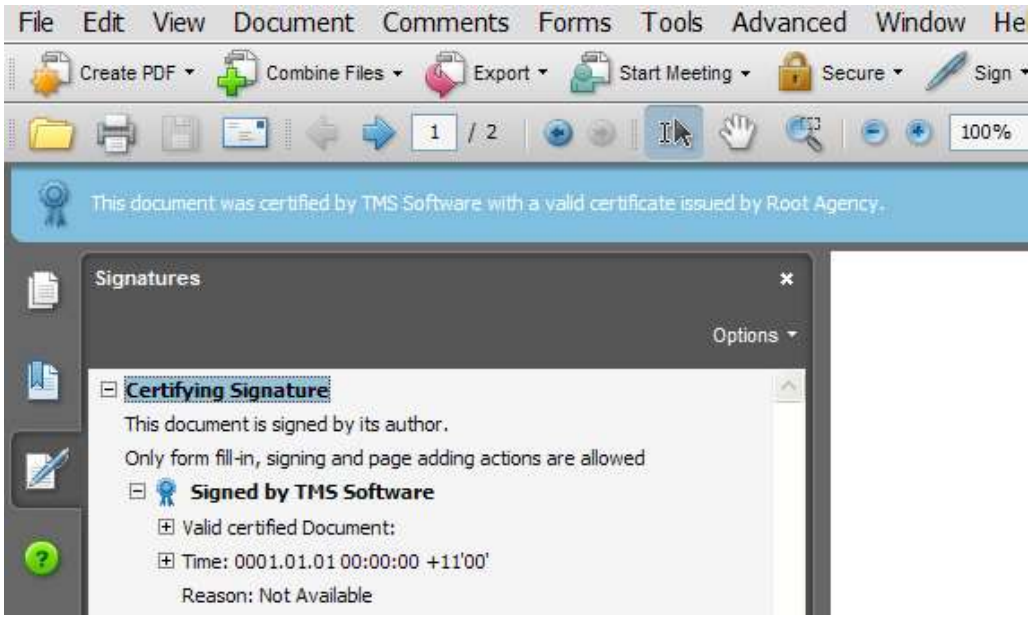
The solution to this problem is to use fonts that include the variants you need.

Normally this is not a problem, since all quality fonts designed to be used with Italics and Bold already come with files for those variants. But if absolutely you need to use fonts that don't come with the variants, you might at least not embed the fonts, so the "fake" style will show.

To be notified whenever FlexCel finds a "fake" style, you can use the "FlexCelError.PdfFauxBoldOrItalics" notifications in TFlexCelTrace.

Signing Pdf Files

FlexCel allows you to sign your PDF files with a certificate, so any change to the file will invalidate it. This is how a signature looks like in Acrobat 8:



In Acrobat 7 you will see the same sidebar at the left, but there is no blue bar at the top.

Important Note: FlexCel signing algorithm is supported only in Acrobat 7 or newer, it will not work in Acrobat 6 or 5 since those versions do not have the required support. Because of this reason, the header in the generated PDF file will automatically switch to say that the file is "Adobe 7 or newer" compatible when you include a signature in your files. If there are no signatures, the default header specifies the file is "Acrobat 5 or newer" compatible.

It is also worth noting that users will still be able to see the generated files in Acrobat 5 or 6, but they will get a warning when opening them and the signature will not validate.

Concepts of signing are outside the scope of this document, but you can find a lot of information in signing in the Acrobat documentation or just in Internet. A good place to start might be:

http://msdn.microsoft.com/msdnmag/issues/07/03/NETSecurity

Export to PDF and FlexCel recalculation

When you create a report with FlexCel most formulas are recalculated, but some are not. This is not an issue when opening the xls file on Excel, as Excel will recalculate the whole file again, but will be an issue when exporting the xls file directly to PDF.

FlexCel implements over 200 Excel functions, and most used formulas are all there so you should not experience big issues. But you need to make sure you do not use any not implemented function to get a properly recalculated sheet. You can find a list of currently implemented functions on the file SupportedFunctions.xls. All functions mentioned there are safe to use.

FlexCel comes with a little utility, the demo "**Validate Recalc**" that will allow you to check if all the formulas on an xls file are ok to use. And of course you can use the code on this demo inside your own application to tell your users when they use a not supported formula.

Using FlexCelPdfExport in OSX and iOS

Since version 6, FlexCel can load the files automatically from memory in OSX and iOS, and it doesn't need access to the font files in disk. So there is no need to define a GetFontFolder event or provide FlexCel a way to get the files.

Note that even when FlexCel will find the fonts automatically, you still need to have the fonts installed in OSX or iOS so FlexCel can use them. In OSX, you will probably want to install the core Windows true type fonts on OSX if they aren't already:

http://corefonts.sourceforge.net/